SOFTWARE ENGINEERING

SOFTWARE EVOLUTION

TABLE OF CONTENTS

[EXECUTIVE SUMMARY 2](#_TOC_250006)

[SOFTWARE EVOLUTION AND WHY IS IT IMPORTANT? 3](#_TOC_250005)

[AGILE METHODS AND EVOLUTION 3](#_TOC_250004)

[SOFTWARE MAINTENANCE 5](#_TOC_250003)

[IS THIS APPROACH TO SOFTWARE EVOLUTION COMPREHENSIVE? 7](#_TOC_250002)

[CONCLUSION 8](#_TOC_250001)

[BIBLIOGRAPHY 9](#_TOC_250000)

TABLE OF FIGURES

FIGURE 1: SPIRAL MODEL 4

FIGURE 2: MAINTENANCE EFFORT DISTRIBUTION 6

## EXECUTIVE SUMMARY

OctoFlex Technologies is a start-up IT firm that is focused on fulfilling all the needs of the businesses. We provide services such as developing a software to reengineering a whole software for small-sized and medium-sized companies. With experienced professionals and a team approach to most of the projects, OctoFlex Technologies will be able to offer a more balanced quality services than many of its competitors in the market.

### The Company

OctoFlex Technologies is a team of two IT professional. Each professional specializes in some particular discipline and has broad knowledge of other disciplines as well. OctoFlex Technologies offers many services whether it is web-based application development or mobile-based application development. The business owners will have a lot of options to select from, depending on their individual needs. This includes developing a website or application on any platform for any business.

### The Market

OctoFlex Technologies promises to provide the best service and customer experience. Its main goal is to fulfill all the needs of its customers. OctoFlex Technologies assures that the final product that it will develop will be of good value.

### Description On Software Evolution

Software Evolution refers to the process of developing the software first, then updating the software for many reasons such as to add new features, fixing bugs, etc. This report outlines the way our software that we have developed will evolve. This report will demonstrate the approach we will use to keep the software running for long time. We will also provide detailed description about different types of software maintenance activities we will do to enhance the software so that it stays competitive and relevant. We will discuss about the reason why this approach to software evolution is comprehensive.

## SOFTWARE EVOLUTION AND WHY IS IT IMPORTANT?

Software Evolution is the term that directly refers to the software system and how this software systems’ get evolved as new requirements are proposed and implemented. The main motive in software engineering is to keep on migrating, upgrading, and evolving the currently in use software systems. Software Evolution is an important part of the software development life cycle. In software evolution, modification in software is required when there are:

* New requirements proposed when the software is being used.
* Changes in business environment.
* New computers and equipment added to the system.
* Defects being reported in the existing software.

Software Evolution plays a vital role in software development life cycle for many reasons which are:

* A large amount of resources are spent on the software by the organizations which makes this software to be a critical business asset. Majority of the software budget is devoted on changing the software rather than developing the new software again.
* To maintain the value of these business assets, new features needs to be added to the software to keep it up-to-date.
* To maintain the value, it is important to continuously keep on improving the performance and reliability of the system.
* Security of the software should also be upgraded from time-to time to keep it safe and relevant.

Software Development and Evolution are the different stages of the software lifecycle which do not differ much from each other. Software evolution follows up after software development that can be shown as:

## AGILE METHODS AND EVOLUTION

Agile methods are based on incremental approach to software development. Evolution is basically a continuation of the development process. It is based on frequent time-to-time system releases.

Changes in the system are expressed as additional user stories that showcases the new additional features of the system on frequent system releases.

There are some handover problems that the development team can face with the evolution team. When the development team uses the agile approach whereas the evolution team uses the plan-based approach, the evolution team expects to have a detailed documentation to

support evolution which is not done in agile processes. If the development team uses the plan- based approach then the evolution team will have to start from scratch to developing an automated test, refactoring of code as it is done in agile development. Automated regression testing is mostly used and is proved to be valuable when changes are required in the system. The solution to this is to use a viewpoint-oriented approach. This will help both the development team and evolution team to work cooperatively.

Usually, Software Development and Evolution are thought to be integrated, iterative process which can be nicely represented through the spiral model.

etc.

Start

*Release 1*

*Release 2*

*Release 3*

Validation

Operation

Implemention

Specification

**FIGURE 1: SPIRAL MODEL**

The process of software evolution is driven by the requests made by the client or the users and includes impact analysis, release planning and change implementation.

There are many older software systems known as legacy systems which are often cheaper and have less risks to maintain which makes them useful for a business. Because they are cheaper and less risker, they are mostly preferred for maintenance than to develop a whole replacement system using the new modern technology. Before, maintenance is started, the business value and the quality of the application will be assessed to finalize if the system should be replaced,

transformed, or maintained. This will help in cost-saving as most of the custom systems that are made usually have the software maintenance cost more than the software development costs.

## SOFTWARE MAINTENANCE

Software maintenance basically means modification of a program after it has already been put into use. It is mostly done for changing custom software. Generic software products are mostly said to evolve over time to create new versions.

After a product has been released, software maintenance is carried out for a number of reasons, including enhancement of the software as a whole, fixing problems or bugs, enhancing performance, and more.

In the SDLC (Software Development Life Cycle), software maintenance comes naturally. Software designers must constantly be looking for ways to correct and improve their work to stay competitive and relevant. They cannot simply launch a product and let it run.

To keep any software running for a long time and satisfy users and customers, using the proper software maintenance techniques and strategies is essential.

However, maintenance continues for at least 15-20 years and sometimes even longer. Full software development takes at least 5 years.

There are commonly three types of software maintenance that are done for software evolution which are:

* **Corrective Maintenance or Fault Repairs:** Corrective Maintenance involves fixing any issues or glitches in the existing software to ensure it functions as intended. This type of maintenance addresses the software's design, logic, code, and potential vulnerabilities. Conducting corrective maintenance can help us identify problems before our customers do, enhancing the reputation of our brand. The corrections usually arise from bug reports submitted by users or customers.
* **Adaptive Maintenance or Environmental Adaption:** Adaptive Maintenance involves modifying the software to adapt to new systems, hardware, software platforms, and changing user needs. This type of maintenance may be necessary due to changes in hardware, software dependencies, cloud storage, or the operating system.
* **Perfective Maintenance (Functionality Addition and Modification):** Perfective Maintenance involves the enhancement of the existing software functionality or the addition of new features to cater to the evolving needs of the users or stakeholders. As the users engage with the software applications, they may identify areas that require improvement or request the inclusion of additional features, leading to future projects or enhancements.

# Fault repair (24%)

Environmental adaptation (19%)

# Functionality addition or modification (58%)

**FIGURE 2: MAINTENANCE EFFORT DISTRIBUTION**

The amount of time spent on each type of maintenance will vary based on the amount of effort spent on each type of maintenance. There are several factors that affect the time to be spent such as the size and complexity of the software, the frequency of updates, and the number of bugs or errors found.

The amount of time spent on each type of maintenance would be:

Corrective Maintenance: Approximately 25% of the total maintenance time is spent on fixing bugs or errors in the existing software to restore normal functionality.

Adaptive Maintenance: Approximately 20% of the total maintenance effort or time is spent on making changes to the software to accommodate new environments, hardware, or software platforms, and changing user requirements.

Perfective Maintenance: Approximately 60% of the total maintenance effort or time is spent on adding new features or improving existing functionality of the software to meet the evolving needs of the users or stakeholders.

Rest 5% of the maintenance time is usually spent on preventive maintenance. **Preventive Maintenance** is also a type of maintenance which is done to prevent unexpected failures from happening in the upcoming future. Mostly this type of maintenance is scheduled and is performed on all the computers and equipment that has been added to the system. Refactoring or making changes in the program to preserve functionality is also a type of preventive maintenance that is used. Refactoring is done to avoid the code degradation that increases the costs and makes the system hard to be maintained in future.

To add an extra effort to make the software easier to be maintained, we usually do software reengineering. **Software Reengineering** is the restructuring or rewriting the part or all of the legacy system without changing its usage or functionality. It can be applied on sub-systems of a larger system that require frequent maintenance.

When setting up a development organization and configuring teams and development methodologies, it is important to consider the amount of time spent on each type of maintenance. A team should be structured in such a way as to allow for efficient handling of bug fixes and adaptations while still allowing time and resources for adding new features and improvements.

## IS THIS APPROACH TO SOFTWARE EVOLUTION COMPREHENSIVE?

Yes, this approach to Software Evolution is Comprehensive because it outlines the overall plan and approach on software evolution. This includes the scope of the evolution process that includes fundamental activities of change analysis, release planning, and system implementation. It states the cost and impact of the changes in the software evolution process that are accessed which affects the change and tells us about how much time and money it will cost to implement these changes. It most importantly outlines the software maintenance activities and its types with the amount of time and effort put into them.

## CONCLUSION

Overall, Software Evolution is the stage in the software system’s life cycle in which software development is already done initially and the software is kept operationally in use for evolvement as new features, or the requirements are proposed and implemented in the system. Evolution is simply a continuation of the development process which is based on frequent software releases. Legacy systems are retained because it is risky and expensive to replace these systems. That’s why system maintenance is done to keep it going for many years to come. There are mainly three type of software maintenance, namely corrective maintenance, adaptive maintenance, and perfective maintenance. Most of the time and effort spent on this type of maintenance. Most of the time spent varies on several factors such as the size and complexity of the software, the frequency of updates, and the number of bugs or errors found. Preventive maintenance is the fourth type of maintenance on which not much of the time and effort is spent but it is essential. Preventive Maintenance includes Refactoring of Code to avoid code degradation so that it is not hard to maintain the software in future. To add some extra effort to make the software easier to maintain in future, Software reengineering is done which includes rewriting part or all of the legacy system without changing its functionality. It is important to consider the amount of time spent on each type of maintenance when setting up a development organization and configuring team and developmental methodologies. A team should be structured in a way that it spends enough time on fixing bugs and repairs and also have enough time to add new features and improvements in the system.

## BIBLIOGRAPHY

*The 4 types of software maintenance & how they help | CAST software***. (**n.d.).

Default. https://[www.castsoftware.com/glossary/Four-Types-Of-Software-Maintenance-](http://www.castsoftware.com/glossary/Four-Types-Of-Software-Maintenance-)

How-They-Help-Your-Organization-Preventive-Perfective-Adaptive-corrective

*4 types of software maintenance & what is software maintenance?* (n.d.). Cloud Protection & Licensing Solutions | Thales. https://cpl.thalesgroup.com/software-monetization/four-

types-of-software-maintenance

*Software engineering | Software evolution***.** (n.d.). GeeksforGeeks | A computer science portal for geeks. https://[www.geeksforgeeks.org/software-engineering-software-evolution/amp/](http://www.geeksforgeeks.org/software-engineering-software-evolution/amp/)

*What makes the software maintenance so important?* (n.d.). APIBEST™ - The first software maintenance company. https://apibest.com/blog/what-makes-software-maintenance-so-

important

*Why is software evolution important?* (n.d.). Educative: Interactive Courses for Software Developers. https://[www.educative.io/answers/why-is-software-evolution-important](http://www.educative.io/answers/why-is-software-evolution-important)